高 等 学 校 教材

C语言程序设计基础辅导教程

赵 宏 编著

钱启平 主审

中 国 铁 道 出 版 社

2000年·北京

（京）新登字 063 号

内 容 简 介

根据教育部对非计算机专业计算机系列课程的要求，铁道部面向 21 世纪教改课题组编写了《C语言程序设计基础》一书。本书是其配套使用的辅导教程，用于辅助学生学习。

全书内容共分三部分：学习要点、上机指导和上机实验。第一部分提供了《C语言程序设计基础》教材各章的内容摘要和部分习题解答；第二部分介绍了 Turbo C 2.0 集成编译环境和 C程序调试和常见错误分析；第三部分根据教学内容安排了十个实验，以培养、训练和提高学习者的程序设计能力和程序调试能力。

本书可作为大专院校非计算机专业计算机基础教育教材和短训班培训教材，也可以作为广大初学者的自学参考书。

图书在版编目（CIP）数据

C 语言程序设计基础辅导教程／赵宏编著. —北京：中国铁道出版社，2000.2
高等学校教材
ISBN 7-113-03692-9

Ⅰ.C… Ⅱ.赵… Ⅲ.C 语言-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2000）第 03505 号

书 名 ： C语言程序设计基础辅导教程
作 者 ： 赵 宏
出版发行：中国铁道出版社（100054，北京市宣武区右安门西街 8 号）
责任编辑：郭 宇 赵静
封面设计：陈东山
印 刷 ：北京市兴顺印刷厂
开 本 ： 787×960 1/16 印张：6 字数：117 千
版 本 ： 2000 年 2 月第 1 版 2000 年 2 月第 1 次印刷
印 数 ： 1～5000 册
书 号 ： ISBN 7-113-03692-9/TP·434
定 价 ： 8.40 元

前　言

本书系“面向21世纪铁路高等教育教学内容和课程体系改革计划项目”，即“非计算机专业计算机基础系列课程教学内容和课程设置改革的研究与实践”课题组组织编写的系列教材之一《C语言程序设计基础》的配套学习辅导书。编写本书的目的就是要帮助读者更好地掌握C语言程序设计的主要内容，掌握程序调试的方法和技巧以及有计划、有步骤地上机实验。

本书包含学习要点、上机指导、上机实验三部分内容。

第一部分学习要点，包括《C语言程序设计基础》各章的内容要点和每章部分习题的解答。本部分简明扼要地说明各章学习的知识点，对每章后的部分难度较大的习题或者容易出错的习题给出了解题思路、算法说明，有的还给出了参考程序。

第二部分上机指导，主要介绍C语言实验的环境和C程序调试的方法和技巧。第一章详细介绍了C语言的集成编译环境Turbo C的使用方法。第二章介绍了C语言程序调试的一般方法与技巧，并对程序中一些常见的错误进行了分析。

第三部分上机实验，包括上机实验安排、要求和实验内容。在第一章中对实验安排做了简单介绍，并提出了程序设计和调试、实验报告等方面应该进行的训练及要达到的要求。第二章是具体实验内容，共安排了十个实验，每个实验都明确规定了实验目的，对每个实验中的题目也都提出了具体的设计和调试的要求，对于较难的题目则给出了解题思路和算法说明。

本书是在作者几年来的教学实践基础上编写的，希望能在教和学两方面都起到积极的作用，但由于编写时间仓促，加之作者水平有限，书中难免有疏漏和不妥之处，恳请读者批评指正。

作　者

1999年12月

第一部分　学习要点

第一章　C 语言和 C 程序概述

1.1　内容摘要

本章重点掌握 C 的程序结构以及上机运行的过程。

1. C语言源程序

用 C 语言所编写的程序称为 C 语言源程序，简称 C 程序。每个 C 语言源程序的文件都必须以 .c 作为文件的扩展名，以说明该文件是 C 语言编写的源程序文件。

2. C语言的程序结构

（1）C 语言程序一般由一个或若干个函数构成，必须有且只能有一个名为 main 的主函数，不管 main()函数放在前或后,程序总是从 main 函数开始执行，在 main 函数结束。被调用的函数既可以是库函数，也可以是自己设计的函数。

（2）函数由函数头和函数体两部分构成，函数说明的一般形式：

```
函数类型　函数名(形参说明)
{
    内部说明语句;
    可执行语句;
}
```

函数头又称函数说明部分，包括函数类型、函数名、形参表和形参说明等，形参可以有也可以没有，但 main 后的（）不能省略。函数体由一对{}括起来，可以有若干个语句，通常由说明语句和可执行语句组成。在以后的程序中我们还会看到这成对出现的花括号{　}还可以用作复合语句的界限符。必须注意，{ 和 }必须成对出现，否则编译时会报告错误。

（3）每行可以写一个或多个语句，一个语句也可以写在多行上，但每个语句必须以分号结束。

（4）可以用/*　*/对 C 中的任意部分做注释，注释在编译时被忽略，即不产生任何代码，它仅仅是帮助人们理解程序。

（5）组成一个程序的若干个函数可以保存在一个或几个源程序文件中，每个文件分别进行编译，然后再连接起来形成可执行文件。

3. C 程序的建立、编译、连接及执行

C 程序与其他高级语言程序一样，必须经过编辑、编译、连接和运行四个步骤如图 1.1 所示。

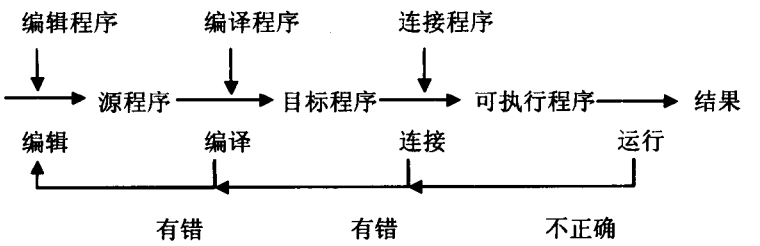

图 1.1　高级语言程序实现的步骤

1.2　部分习题解答

5. 参考【例 1.1】，编制一个完成对三个数作相加和相乘操作的程序。

```
/* 主函数 */
main()
{    int a,b,c;
     printf("请键入三个整数\n");
     scanf("%d%d%d",&a,&b,&c);
     add(a,b,c);
     mul(a,b,c);
}
/* 对三个整型数做加法则并输出显示结果 */
add(int x,int y,int z)
```

```
{   int sum;
    sum=x+y+z;
    printf("%d+%d+%d=%d\n",x,y,z,sum);
}
 /* 对三个整型数做乘法则并输出结果显示 */
mul(int x,int y,int z)
{   int fac;
    fac=x*y*z;
    printf("%d*%d*%d=%d\n",x,y,z,fac);
}
```

第二章　基本数据类型

2.1　内容摘要

程序操作的对象是数据，数据及有关的概念和知识都是最基本的，应该深入理解和掌握。

1. 数据

数据是程序处理的对象，它被划分成不同的类型。如 int（整型）、float（单精度浮点型）、double（双精度浮点型）、char （字符型）等。各种数据类型的数据都有其确定的取值范围和能参与的运算，例如，在 int类型的数据上可以进行＋(加)、－(减)、*(乘)、/(除)、％ (求余）等运算。因此，在学习一种数据类型时，要把这种数据类型的数据书写规则和可以进行的运算联系起来学习。

C语言除提供以上几种基本数据类型外，还允许用户定义其他基本数据类型（如枚举型)、构造类型(如数组类型、结构体类型、共用体类型)、指针类型和空类型。

2. 常量和变量

数据有常量和变量之分，常量和变量是程序中使用的最基本的数据对象。

常量是程序执行过程中不能改变的量。有些类型的常量从字面形式就可以判别出来，如 123 为整型常量，0.65 为实型常量，'A'为字符型常量，"CHINA"为字符串常量。

要仔细区分字符常量和字符串常量。字符常量是用单引号括起来的一个字符，实际存储的是机器字符集中字符的 ASCII 值。字符串常量是由一对双引号括起来的字符序列，是包含转义字符“ \0 ”的一个字符数组。

C 语言规定也可以用标识符代表常量，称为符号常量。如：

```
#define PI 3.14159      /*  定义符号常量 PI 的值为 3.14159*/
```

变量是存放数据的存储空间，空间大小视变量的数据类型而定，在程序执行过程中变量的值是可以改变的。使用变量定义，可以设置程序中使用的存储空间，每个存储空间有个名字和存放数据值的类型。

2.2 习题二部分习题解答

2．字符常量和字符串常量有什么区别？

解答：

字符常量是用单引号括起来的一个字符，字符串常量是由一对双引号括起来的字符序列。字符常量占用内存一个字节，实际存放的是该字符在机器字符集中的 ASCII 码值，字符串常量占用内存中地址连续的若干个字节，字符串在存储时末尾被系统自动加上转义字符“ \0”。

4．写出下面程序的输出结果。

（3）

```
main( )
{    char c1='a',c2='b',c3='c',c4='\101',c5='\116';
     printf("a%cb%c\tc%c\tabc\n",c1,c2,c3);
     printf("\t\b%c   %c",c4,c5);
}
```

提示：在 TurboC 中，系统把一行分为 10 个输出区，每个输出区占 8 列。“\t’,的作用是跳过一个输出区，“\b ”的作用是向回跳一格。

解答：aa　　bb　　cc　　abc

　　A　N

第三章　运算符和表达式

3.1　内容摘要

1. 运算符

C 语言提供了许多运算符，学习时，主要掌握这些运算符的运算规则、优先级和结合性。

（1）算术运算符

＋（加法运算或正号运算符）

—（减法运算或负号运算符）

（乘法运算符）

／（除法运算符）

％（求余运算符，或称模运算符）

算术运算符的结合性是“从左到右”。

（2）自增和自减运算符

C 语言提供了使变量自增自减的运算符++和一。自增运算符的作用是使变量的值增 1，自减运算符的作用是使变量的值减 1。

自增和自减运算符的结合方向是“自右向左”。如表达式-i++等价于-(i++)。

（3）赋值运算符

C 语言中“=”就是赋值运算符，它的作用是将一个数据赋值给一个变量。如“a=5”的作用是执行一次赋值操作（或称赋值运算），把常量 5 赋给变量 a.。也可以将一个表达式的值赋给一个变量，如 x = 3 + 5 * 6。这些属于简单的赋值运算符。在赋值符“=”之前加上算术运算符，可以构成复合的赋值运算符。例如：

i + = 2　　等价于 i = i + 2

a * = b + 5　　等价于 a = a * (b + 5)

x% = 3　　等价于 x = x%3

赋值运算符按照“自右向左”的结合顺序，因此“a=b=8”相当于“a=(b=8)”。

（4）关系运算符

C 语言提供 6 种关系关系运算符：

<　　（小于）

<= （小于等于）

> （大于）

>= （大于等于）

== （等于）

！= （不等于）

关系运算符的结合方向“自左向右”。

（5）逻辑运算符

C语言提供三种逻辑运算符：

&& （逻辑与）

|| （逻辑或）

！（逻辑非）

逻辑运算符的结合方向是“从左向右”。

（6）条件运算符

C语言中只有一种三目运算符，即“？：”。它只有如下形式：

表达式1？表达式2：表达式3

这就是条件表达式，先计算表达式1，若其值为非零，则结果为表达式2的值，否则就是表达式3的值。

（7）逗号运算符

表达式1，表达式2，…，表达式n

由逗号隔开的一组表达式从左向右进行计算，其求解过程为：先求解表达式1，再求表达式2，…，整个逗号表达式的值是表达式n的值。

2. 运算符的优先级

!	（非运算符）	高
++，--	（自增自减运算符）	
*，/，%	（算术运算符）	
+，-	（算术运算符）	
<，<=，>，>=	（关系运算符）	
==，!=	（关系运算符）	
&&	（逻辑与运算符）	
||	（逻辑或运算符）	
?:	（条件运算符）	
=，+=，-=，*=，/=，%=	（赋值运算符）	
,	（逗号运算符）	低

3. 表达式

在程序中，凡是表示一个值或要计算一个值的地方就要用到表达式，用运算符把运算量连接起来形成一个有意义的式子叫表达式。

表达式中可以使用小括号，因此用括号括起来的表达式是有层次的，在求表达式的值时，要从内层往外层（即先计算内层括号，后计算外层括号），同一层的表达式，从左到右，按照优先级的高低依次计算，如果运算量两侧的运算符优先级相同，则按规定的结合方向进行。

3.2 习题三部分习题解答

2．写出下面逻辑表达式的值，设 a=3,b=4,c=5。

a+b>c&&b=c

a||b+c&&b-c

!(a>b)&&!c

!(x=a)&&(y=b)&&0

!(a+b)+c-1&&b+c/2

提示：C 语言中用非 0 代表真，用 0 代表假。如果一个表达式的值为真，则结果用 1 表示，表达式的值为假，结果用 0 表示。

解答：（1）1　　（2）1　　（3）1　　（4）0　　（5）1

4．写出下面程序的输出结果。

```
main()
{   int x=1,y=1,z=1;
    y=y+z;
    x=x+y;
    printf("%d\n",x<y?y:x);
    printf("%d\n",x<y?x++:y++);
    printf("%d\n",x);
    printf("%d\n",y);
    x=3;
    y=z=4;
    printf("%d\n",(x>=y>=x)?1: 0);
    printf("%d\n",z>=y&&y>=x);
}
```

提示：阅读本题时要注意运算符的优先级和结合方向。

解答：3

2

3

3

0

1

第四章　程序的控制语句

4.1　内容提要

1. 函数调用语句

函数调用语句由一次函数调用加上一个分号构成，调用的函数既可以是自己定义的函数，也可以是 C 语言的库函数。如：

```
y=sqrt(x);
```

调用了 C 语言的求开平方的库函数 sqrt()。

C 语言中比较基本也比较常用的是基本输入／输出函数。C 语言不提供输入输出语句，但在 C 语言的标准函数库中，定义了一些输入输出函数，通过调用这些函数可以实现数据的输入输出。目前只使用键盘读入数据、往显示器上输出数据。

（1）字符输出、输入函数

字符输出、输入函数一次只能输出输入一个字符。要注意三个字符输入函数 getchar()、getch()或 getche()的差别：

getchar() 函数在由键盘键入一个字符后，必须按回车键，而 getch()和 getche()函数不需要，getche()函数和 getchar()会显示出所输入的字符，而 getch() 函数不显示输入的字符。

（2）格式输出、输入函数

输入数据时，从左到右，每读一个数据，就相应地存放进格式输入函数的地址列表中。地址所确定的变量，要注意读入数据和地址列表中地址的对应关系。输出数据时，把格式输出函数中的表达式的值从左到右逐个写到显示器上，并用格式控制符控制书写格式。

2. 赋值语句

赋值语句是由赋值表达式加上一个分号构成，作用是将一个确定的值赋给一个变量。格式为；

变量名=表达式;

赋值语句有如下特点：

（1）先计算，后赋值。即先计算右边表达式的值，然后将该值赋给赋值号左边的

变量。程序中的求值计算主要是用赋值语句来实现的。

（2）赋值语句中的“ = ”是赋值号而不是数学意义上的等号，在数学上 i=i+1 是不成立的，而程序设计中是允许的。赋值号两侧的内容不能任意调换。

3. 实现选择结构语句

（1）if 语句

if 语句的语法：

```
if(表达式)  { 语句 1;}
[ else {语句 2;} ]
```

注：方括号括起的部分可有可无，下同。

if 语句的语义 若表达式的值非零(真),则执行语句 1，然后跳过 else；如果表达式的值为 0(假),则跳过语句 1,执行语句 2。如果表达式的值为 0 时不执行任何操作，则可以省略方括号括起的部分。

在 if语句中“ 语句 1 ”和“ 语句 2 ”部分可以是一个语句，也可以是一组语句。如果是一个语句，则可以省略花括号。“ 语句 1 ”和“ 语句 2 ”部分可以是任何一个语句，当然也可以又是一个 if 语句，这种情况称为 if语句的嵌套。在使用 if 语句的嵌套时，要特别注意 if和 else 的配对关系，特别是当 if和 else 数目不同时。C语言规定，else子句总是和它前面离它最近的没有 else 子句的 if 子句配对。

（2）switch 语句

switch 语句的语法：

```
switch (表达式)
{       case 常量表达式 1：语句 1;[break;]
        case 常量表达式 2：语句 2;[break;]
        …
        case 常量表达式 n：语句 n;[break;]
        default            ：语句 n+1;
}
```

switch 的执行过程是，先计算 switch后面表达式的值，然后依次与关键字 case后的常量表达式相比，当它们一致时就执行该 case后的语句，若表达式的值与所有 case后常量表达式的值都不同，则执行 default 后的语句。在执行某个语句时遇到 break 语句，则退出 switch 结构。

在 switch 结构中：

①switch后括号的表达式的值必须是整型，它经常是一个整型变量。

②各个 case 分枝的常量表达式必须是整型常量，它可以是一个整数、一个字符常量或一个整型常量表达式。

③两个 case 分枝的常量表达式的值不能相同，否则执行程序时不知选择哪个常量表达式后面的语句。

4. 实现循环结构语句

（1）三个循环语句的语法和语义

①while 语句用来实现“当型”循环，基本形式为：

while(表达式) 语句；

其执行过程是首先判断表达式，当表达式的值为 0 时，退去 while 循环，执行 while 语句的下一语句；否则，当表达式的值为非 0 时，执行“语句”后，再次执行该 while 语句。

②do—while 语句用来实现“直到型”循环，基本形式为：

do 语句

while(表达式);

其执行过程是重复执行循环体“语句”，在每次执行“语句”后，都测试表达式的值，直到表达式的值为 0 时退出循环。

③for 语句的一般形式为：

for(表达式 1；表达式 2；表 达 式 3) 语句；

其执行过程是，首先处理表达式 1，再测试表达式 2 的值，若其值非 0，则执行循环体语句，最后处理表达式 3。至此完成了一次循环，然后再测试表达式 2 的值，直到表达式 2 的值为 0(假)，则退出循环。

（2）三个循环语句中，while 是最基本的，凡是能用 do—while 语句和 for 语句编写的程序，都能用 while 语句编写出来，但对某些问题，使用 do—while 语句和 for 语句编写程序更简单。

一般来说，在不知道数据的数量或不知道重复计算的次数时，如用迭代法对非线性方程求解，用 while 语句和 do—while 语句。与上述情况相反，如果知道数据的数量或知道重复计算的次数，例如对数组进行处理，一般用 for 语句，并且用循环控制变量作为数组的下标有很强的表达能力。

5. 从实际例子中深刻理解循环结构

循环结构的实际例子非常多，读者深入理解下面几个例子是有帮助的。

若干项累加：sum=a0+a1+a2+…+an+…

当|an|≤eps 时，即达到一定精度时舍去。用 while 表示如下：

```
sum=a0;
n=1;
计算 an;
```

```
while (fabs(an)>=eps)
{    sum=sum+an;
     n=n+1;
     计算 an;
}
```

针对不同的问题，利用上述算法编写程序时，只需在计算 an 时用具体的项代入即可，如已知 $e^x=1+x/1!+x^2/2!+\cdots+x^n/n!+\cdots$ ，对给定的 x，求 e^x 值，程序如下：

```
#include"math.h"
main()
{    int n=1;
     float sum=1,an,x;
     scanf("%f",&x);
     an=x;
     while   (fabs(an)>=1e-6)
     {    sum=sum+an;
          n=n+1;
          an=an*x/n;              /*计算 an*/
     }
printf("The result is %f",sum);
}
```

（2）百钱买百鸡问题。公鸡每只 5 元，母鸡每只 3 元，小鸡 3 只一元，问一百元买一百只鸡有几种解法。它的基本思路是：一一列举各种可能的情况，并判断哪一种情况是符合要求的解。

枚举出 x,y,z 的所有可能组合，选取满足条件的组合。用 for 语句表示如下：

```
for(x=0; x<=100; x++)
for(y=0;y<=100-x; y++)
{   z=100-x-y;
      if (x,y,z 的取值满足条件)
      printf(输出一组 x,y,z 组合);
}
```

经过分析优化，上述算法可表示为：

```
for(x=0; x<=19; x++)
for(y=0;y<=33-x; y++)
{   z=100-x-y;
    if (x,y,z 的取值满足条件)
```

```
        printf(输出一组 x,y,z组合);
    }
```

与百钱买百鸡问题类似的问题，如用一张百元币，换成 5 元、1 元和 5 角币共 100 张，且每种不少于 1 张。可以仿照上面的例子进行程序设计。

4.2 习题四部分习题解答

1. 阅读程序，写出运行结果。

（1）

```
#include "stdio.h"
main()
{   char c;
    while((c=getchar())!='\n') printf("%4d",c);
    printf("\n");
}
```

解答：如果输入任意一个非回车字符，则输出其相应 ASCII 码，然后输出一个换行符。如果输入一个回车符，则只输出一个换行符。

（2）

```
#include "stdio.h"
main()
{   char c;
    while(c=getchar()!='\n') printf("%4d",c);
    printf("\n");
}
```

解答：如果输入任意一个非回车字符，则输出数值 1，然后输出一个换行符。如果输入一个回车符，则只输出一个换行符。

此题的关键是 while后的表达式，“ ！= ”的优先级高于“ = ”,所以键入一个字符后，首先判断此字符是否不等于“ \n”，如果为真，则变量 c被赋值为 1，即 while后的表达式为真,则执行循环体,否则变量 c 被赋值为 0，while后的表达式为假，不执行循环体。

3. 编写程序，输入三角形三边 a、b、c,判 断 a、b、c 能否构成三角形，若不能则输出相应的信息，若能则判断组成的是等腰、等边、直角还是一般三角形。

解答：a、b、c 能否构成三角形的条件是“ 任意两边之和大于第三边 ”，即